

21
150 pts.

ARUN

Enciclopedia Práctica del Spectrum



Nueva Lente/Ingelek





ATRIBUTOS DE IMPRESION



Tenemos en cuenta el hecho de que los colores pueden adaptar el brillo normal y al especial, podemos decir que contamos con una paleta de diecisiete colores para trabajar, siempre dentro del área imprimible de la pantalla, puesto que para el mismo sólo es posible el empleo de los 8 bits.

Esto nos sugiere la idea de crear una tabla de color, compuesta por el código de dato y su brillo, asociado a un nombre de color concreto, tal como se detalla en la tabla adjunta.

Esta tabla es igualmente aplicable a la sentencia PAPER y a la INK, puesto que se trata de un atributo de la posición de pantalla. De esta forma, cuando es necesario imprimir en un lugar concreto de la misma, el Sistema analiza los colores asignados a la posición, y utiliza para los puntos iluminados el que indica INK y para los no iluminados el de PAPER, quedando los dos asociados por el brillo de la posición (normal o suplementario).

Sabiendo esto, es fácil comprender que FLASH se limita a alterar el contenido de los atributos de tinta y color, de forma intermitente, consiguiendo el efecto parpadeante en la posición o posiciones de pantalla afectadas.

TABLA DE COLOR

CODIGO	NOMBRE	COLOR SIN BRILLO	COLOR CON BRILLO
0	Black	Negro	Negro resaltado
1	Blue	Azul marino	Azul marino claro
2	Red	Rojo oscuro	Rojo claro
3	Magenta	Rosa oscuro	Rosa claro
4	Green	Verde oscuro	Verde claro
5	Cyan	Azul celeste	Azul celeste claro
6	Yellow	Amarillo oscuro	Amarillo claro
7	White	Gris claro	Blanco

ninguna sentencia de color. Realmente es así, aunque en determinadas ocasiones simplifica la programación.

Veamos un ejemplo. Supongamos que al comenzar

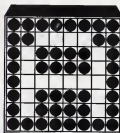
En la pantalla del ordenador, los puntos encendidos se representan en el color de primer término (INK) y los apagados en el de fondo (PAPER).

CODIGOS DE COLOR ESPECIALES

Los códigos especiales son los valores 8 y 9 de los parámetros de color. Al ser números mayores que siete, es evidente que no pasan a formar parte directa del atributo de color de la posición, sino que señalan al Sistema determinada forma de interpretación.

El código 8 indica color TRANSPARENTE. Se trata, pues, de un código de color neutro, que no altera en absoluto el elegido anteriormente para la posición, pudiendo emplearse con PAPER, INK, BRIGHT y FLASH.

En principio, podría parecerse que el uso de este código de control es lo mismo que no emplear



!

El juego completo de colores del Spectrum, combinando las dos intensidades de brillo, la integran de esas tonalidades diferentes.

*

Los parámetros utilizados para los códigos de color van desde el cero hasta el ocho, más dos valores especiales: el ocho y el nueve correspondientes a los niveles de transparencia y contraste, respectivamente.

*

El código de transparencia no afecta el color de las pautas, sino a los cuartos de tinta.

*

El código de contraste equivale a siete (blanco) o cero (negro), según las necesidades tendientes a obtener una máxima legibilidad de los caracteres oscuros.

zo de un programa, hemos asignado a la pantalla un determinado color de fondo; a continuación, unas instrucciones más adelante, tras haber utilizado reiteradas veces sentencias INK y PAPER, condicionadas mediante IF o por cualquier otro sistema, no tenemos conocimiento del color de fondo existente actualmente en la pantalla. Si quisiéramos ahora escribir un carácter en dicho color de fondo, nos sería imposible, dado que no sabríamos qué parámetro especificar para PAPER. Pues bien, indicando el código 8, le encargamos al ordenador que sea él el que averigüe el color actual de la pantalla, escribiendo el carácter sin modificar dato.

El código 9 corresponde al color de CONTRASTE, siendo únicamente aplicable a los atributos PAPER e INK. Este selecciona, cuando se aplica, el color adecuado del atributo que se desea modificar en función del valor del otro, de forma que se obtenga la máxima visibilidad.

Con esto conseguimos que el Spectrum asigne, INK 0 (negro) para los PAPER de tonalidad clara o INK 7 (blanco) para los de tonalidad oscura. Para hacer esto, el ordenador sabe que los códigos de color del 0 al 3 corresponden a los tonos oscuros, y los comprendidos entre el 4 y el 7 a los claros.

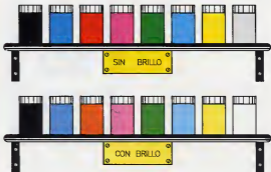
De manera similar, al emplear PAPER 9, el Spectrum asigna los códigos de fondo del papel



El código de color ocho sirve para tinta transparente.

de forma que presenten un contraste con la tinta existente en esa posición, es decir, fondo negro para los tintas claras, y blanco para las oscuras. La utilización del código de control 9, garantiza que los caracteres que escribamos sean siempre legibles. No obstante, conviene saber que este código emplea tan solo los tonos blanco y negro, por lo que si nos referimos a los dos atributos de determinada posición con este mismo código, es

Maximizando convenientemente la posibilidad de brillo y transparencia, nuestro Spectrum dispone de un total de dieciséis colores.





El código de color hace evidente el punto de mayor contraste, utilizando para ello los colores blanco a negro, negro a blanco.

dear, asignando antes de escribir un carácter tanto PAPER 9 como INK 9, obtendremos negro sobre blanco (estado al conectar el ordenador), o blanco sobre negro (video invertido).

Veremos más claras estas combinaciones en el programa de ejemplo titulado EFECTOS CODIGO 9

La generación de los ocho colores básicos del Spectrum se fundamenta en la combinación de los tres primarios: rojo, azul y verde.



```

10 REM - EFECTOS CODIGO 9
20 INK 9
30 FOR J=0 TO 7
40 PAPER 1
50 FOR J=0 TO 1
60 BRIGHT J
70 PRINT "PAPER "J;" "
80 FOR K=0 TO 23: PRINT "K";: NEXT K
90 NEXT J: NEXT I
100 BRIGHT 9
110 PRINT ""TAB 4;"COMBINACIONES CON I
    NK 9"
120 PAUSE 500: CLS
130 PAPER 9
140 FOR I=0 TO 7
150 INK I
160 FOR J=0 TO 1
170 BRIGHT J
180 PRINT "INK "J;" "
190 FOR K=0 TO 25: PRINT "K";: NEXT K
200 NEXT J: NEXT I
210 PAPER 7: INK 9: BRIGHT 9
220 PRINT ""TAB 3;"COMBINACIONES CON P
    APER 9"

```

OBTENCION DE LOS COLORES BASICOS

Ya hemos dicho que los colores básicos son los ocho que se corresponden con las teclas del 0 al 7, en la fila superior del teclado. En cualquier caso, estos colores básicos no son más que combinación de tres fundamentales, sobre los cuales se desarrolla la generación del color en los receptores de TV.

Estos tres colores realmente fundamentales son el azul, el verde y el rojo. Por combinación de ellos, se obtienen todos los demás.

Si somos un poco observadores, encontraremos unas relaciones muy interesantes entre los códigos y los colores. La combinación de azul (código 1), con el rojo (código 2), nos da el color magenta (código 3). Del mismo modo, la combinación del color rojo (código 2), con el verde (código 4), nos da el color amarillo (código 6). Y así sucesivamente, los códigos van coincidiendo con la suma de los colores que los componen.

Por supuesto, esta relación de suma no es una pura casualidad. Como norma general, podemos decir que los 8 colores básicos del Spectrum, aparte del azul, el verde y el rojo, se corresponden con las combinaciones de los diferentes pares de los fundamentales entre ellos.

Para hacernos una idea más gráfica de todo esto, podemos imaginar un escenario oscuro (obtenido por ausencia de todos los colores), sobre el cual

!

Con los botones **INK**, **PAPER**, **BRIGHT** y **FLASH**, se pueden escribir los parámetros 1 y 6, pero su operación y desactivación respectivamente, así como el código 8, utilizando para el efecto de transparencia.

*

La zona de la memoria en la cual el ordenador almacena los atributos correspondientes a cada carácter de la pantalla, se denomina **ZONA DE ATRIBUTOS**.

*

En la **ZONA DE ATRIBUTOS** se fijan los parámetros de estado de **INK**, **PAPER**, **BRIGHT** y **FLASH** de cada posición de carácter de la pantalla.

*

La instrucción **OVER** 1 nos da acceso a la superposición de caracteres.

dirigimos tres haces de luz con los colores azul, verde y rojo. El resultado de este curioso experimento sería más o menos el que se obtiene mediante el programa titulado OBTENCION DE LOS COLORES BASICOS

```

10 REM = OBTENCION COLORES BASICOS
20 SCRIBE 0: PAPER 0: INK 7: CLS
30 PRINT "OBTENCION DE LOS COLORES BAS
40 ...
50 FOR K=190 TO 150 STEP 10: GO SUB 70
60 NEXT K
70 PRINT AT 4,9: "A" AT 10,8: "A" AT 16,
14: "A" AT 20,20: "A" AT 16,14: "A" AT 19,2
: "A" AT 19,20: "A"
80 STOP
90 FOR I=0 TO 5
100 RESTORE K: PRINT " "
110 FOR J=0 TO 4: READ X: PRINT INK X:
: " " NEXT J
120 PRINT
130 NEXT I
140 RETURN
150 DATA 0,2,2,2,0
160 DATA 1,3,7,6,4
170 DATA 1,1,5,4,4
    
```

De esta forma se generan los 8 colores básicos. Los tres primeros por ser primarios (esta es la denominación técnica que reciben los mencionados tres colores fundamentales), tres más como combinación directa de ellos tomados dos a dos, y uno más, el blanco, por combinación de los tres fundamentales.



El efecto de INVERSE 1 en la pantalla es similar a la acción de un negativo fotográfico.



LA SENTENCIA INVERSE

La zona de la memoria donde el ordenador almacena la disposición de los puntos encendidos y apagados en la pantalla, el margen de los avisos, se denomina ARCHIVO DE IMAGEN.



Cuando un carácter se escribe en OVER 1 dos veces consecutivas sobre una misma posición, se recupera el carácter primero sobre el cual se superpone.

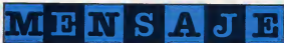
Cada carácter representado en la pantalla, se representa dentro de un cuadrado (posición de carácter) configurado por 64 puntos, distribuidos en 8 filas de 8 columnas. Dentro de esta estructura reticular, los puntos iluminados adoptan el color del atributo INK, y los no iluminados el de PAPER.

Hemos visto también, que el efecto de FLASH se intercambia los valores de INK y PAPER, de forma intermitente, para conseguir una antinaturalidad.

INVERSE hace algo muy parecido, aunque de una forma permanente: intercambia los valores de INK y PAPER de determinada zona de la pantalla, para obtener lo que se denomina comúnmente como vídeo inverso, similar a lo que vemos en un negativo fotográfico.

INVERSE tiene también un paralelismo evidente en su utilización con BRIGHT y FLASH, puesto que existe también un INVERSE 1 y un INVERSE 0. INVERSE 1 marca el inicio de la actividad de esta función de impresión, e INVERSE 0 determina el final.

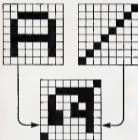
La combinación de FLASH e INVERSE nos permite mostrar de una manera muy original los caracteres en la pantalla.



En cualquier caso, conviene que nos fijemos en que esta función de impresión no afecta a los atributos de una posición determinada de la pantalla, del mismo modo en que lo hacen las opciones anteriormente descritas.

En este caso, se trata de una opción que califica a la totalidad de la impresión, pero sin dejar huella en la zona de atributos de las posiciones de pantalla.

La función OVER no ejerce directamente una acción sobrecentada.



La sentencia OVER establece la posibilidad de sobrecentar.

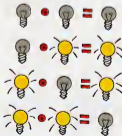
```
10 REM INVERSE / FLASH - J.M. LOPEZ MARTINEZ
20 INPUT "Titulo": LINE X0: IF X0="" THEN STOP
30 LET X=LEN X0: IF X<=32 THEN GO TO 40
40 CLS: FLASH 1
50 FOR I=1 TO X
60 PRINT AT 11,(32-X)/2+I-1, INVERSE I/2-INT(I/2):X0(I)
70 NEXT I
80 FLASH 0: GO TO 20
```

Aunque más adelante trataremos en profundidad esta toma, haremos ahora una pequeña aclaración al respecto. Una zona de la memoria del ordenador, denominada ZONA DE ATRIBUTOS, es utilizada por éste para almacenar los códigos de color que corresponden a cada una de las posiciones de la pantalla.

Cuando ejecutamos una sentencia FLASH, BRIGHT, INK o PAPER, los estados que indican quedan reflejados en dicho área de la memoria. Sin embargo, al ejecutar una instrucción del tipo INVERSE 1, el ordenador se limita a transponer directamente los datos de INK y PAPER en las siguientes impresiones, pero no reflejando en la zona de atributos si esto se debía a una sentencia INVERSE, sino simplemente como si el programador hubiera cambiado los códigos de color mediante INK y PAPER.

Puesto que la utilización de INVERSE es bastante frecuente, mostraremos un ejemplo de ello. El siguiente programa imprime el mensaje que le indiquemos, carácter a carácter, centrado en la pantalla, haciendo uso de la conexión y desconexión de la función de impresión INVERSE para producir un resaltado más original que el de FLASH.

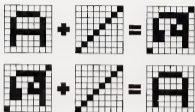
OVER ejerce una operación lógica XOR (o exclusiva, que se representa mediante un signo menos encerrado en un círculo), entre los puntos de los caracteres a representar.



Las declaraciones de atributos de color combinadas con la sentencia PRINT en forma de función, tienen un valor temporal, debido a su mayor prioridad que el permanente.



Las sentencias de atribución de color utilizadas como palabras clave, tienen un valor permanente, aunque para todas las impresiones sucesivas siempre y cuando estas no sufran atribuciones temporales.



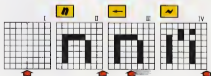
Una propiedad de OVER es la vuelta al carácter original cuando se escribe de nuevo el supercarácter.

LA SENTENCIA OVER

Del mismo modo que INVERSE nos da paso a la posibilidad de impresión en vídeo inverso, OVER nos permite acceder a la sobreescripción de caracteres.

OVER tiene también conexión y desconexión de la forma explicada anteriormente, es decir, por medio de OVER 1 y OVER 0, respectivamente. Asimismo, esta sentencia no tiene influencia permanente sobre la configuración de los atributos de pantalla de las zonas a las que afecta (presencia en el área de atributos). Se trata, como en el caso de INVERSE, de una posibilidad de impre-

Para formar los caracteres castellanos así aparecen en la función OVER y el carácter de control para retroceso (CTRL-B).



sión externa, no controlada por los atributos. Cuando escribimos normalmente sobre la pantalla, cada nuevo carácter que coincide en una posición ocupada ya por otro, reemplaza al antiguo. Este hecho no debe importarnos normalmente, pero nos impide la sobreescripción, es decir, la posibilidad de acceder a caracteres especiales como composición de dos o más simples.

En el caso de los programadores hispanoparlantes, esta facultad de impresión es de especial importancia, dado que nos permite obtener caracteres propios del Castellano, como la eñe (ñ) o las vocales acentuadas, que en un principio nos están vedados a los usuarios del Spectrum, puesto que no son caracteres disponibles en el alfabeto inglés.

Dado el interés de esta aplicación concreta de OVER, observaremos un programa de ejemplo, que ilustra la forma de conseguir los caracteres castellanos antes mencionados.

```
10 REM - OVER
20 OVER 1
30 PRINT AT 9,5;"Bajo el teclado de
  los..."
40 PRINT "Reyes Cató"OVER 0;"licos,
  de castiella"OVER 0;"la..."
50 PRINT "unso"OVER 0;"n de Espa"OVER 0
  "RS 0;"a y Portugal."
60 OVER 0
```

Para acceder a la escritura de estos caracteres dobles, empleamos un método bastante simple: en primer lugar, escribimos la letra básica, como ya sabemos, esto produce el avance a la siguiente posición de escritura. Así pues, deberemos retroceder, para volver a situarnos sobre la letra básica, y aprovechándonos de OVER, escribir sobre ella, sin borrarla, el signo complementario deseado.

Así pues, obtendremos la eñe (ñ), escribiendo una eñe (e), retrocediendo y escribiendo la línea superior que le caracteriza (ñ). De modo similar, podremos obtener las vocales acentuadas, mediante la impresión previa de la vocal, al retroceso de un espacio y la sobreescripción de un carácter de acentuación (´), obtenido mediante SYMBOL SHIFT + 7.

De todo este proceso, el único elemento que todavía nos es ajeno es el sistema cómodo para retroceder el paso dado durante la primera impresión. Hicimos poco que adelantamos el estudio del código A 5 C I, de caracteres, y en él se puso de manifiesto que los 32 primeros caracteres (del 0 al 31), carecen de una representación en la pantalla. Estos son los denominados códigos de control, que no representan ningún carácter, sino que efectúan una determinada función.

Uno de estos códigos de control, concretamente el ocho, es el que realiza la función de retroceder una posición de carácter el puntero de escritura, se encuentre donde se encuentre actualmente. En lo referente a su obtención, y puesto



El carácter de control 5 (CHR\$ 5) produce el retroceso del punto de impresión en una posición, se enciende donde se encuentra

que conocemos su código, utilizaremos la función CHR\$, cada una hace apenas unas páginas, que nos permite obtener un carácter en base a su código (PRINT CHR\$ 8).

Una cosa más debemos conocer sobre OVER, relativa a su forma de activar. Cuando se emplea esta función de impresión, se sobrepone el segundo carácter sobre el primero, pero de forma que donde coinciden dos puntos iluminados, se sólo un punto sin iluminar (del color del papel). Este hecho, que en principio puede parecer algo extraño, nos permite recuperar el carácter primitivo, repitiendo la operación de sobrepresión con OVER 1. Como prueba de ello, podemos ver lo que sucede al ejecutar esta línea de programa:

```
10 CLS: OVER 1: PRINT "0";CHR$ 8; PAU
SE 250: PRINT "8"; GO TO 10
```

Este resultado corresponde con el de una función Maja, denominada XOR, que será estudiada junto con otras funciones de este tipo en las próximas páginas dedicadas a la sección TU SPECTRUM.

ATRIBUTOS TEMPORALES

Hasta ahora hemos visto que los atributos afectan de forma permanente a las impresiones su cosas, hasta que no se recibe una nueva instrucción para alterar este estado de cosas. En ciertas ocasiones, es muy interesante depen-

der de una asignación temporal de los colores. Por ejemplo, podemos estar escribiendo habitualmente en blanco sobre azul, e interesarnos destacar las palabras oscure de los párrafos en amarillo. Estar cambiando de color cada vez que se escribe el comienzo de párrafo, la primera vez para asignar el amarillo, y a continuación para restablecer el blanco habitual, puede resultar ciertamente engorroso. Para superar este problema, el Spectrum dispone de un sistema de asignación temporal de los atributos.

Esto consiste en utilizar las palabras BASIC para control de atributos, en conjunción con una sentencia PRINT. Así por ejemplo, si nosotros escribimos INK 4: PRINT "A", la letra A será escrita en verde, como pretendíamos, pero también todos los caracteres que se escriben desde ese momento, hasta la ejecución de otra sentencia INK, se representarán en el mismo color.

En vez de esto, podemos sustituir la instrucción del ejemplo por PRINT INK 4;"A". En un principio, el efecto parece idéntico, pero pronto comprobaremos que no es así, puesto que los caracte-

Mediante las teclas 4 y 3, en combinación con CAPS SHIFT, podemos acceder a los códigos de control de INVERSE.



CAPS SHIFT + 4

CAPS SHIFT + 3

BITS

El estado de OVER o INVERSE no es representado en la serie de atributos, sino que afecta directamente al ancho de imagen (en el primer caso), y a los colores de INK y PAPER simultáneamente, en el segundo.

*

Los televisores obtienen cualquier color en base a la combinación de los tres fundamentales: rojo, azul y verde.

*

Los códigos que reciben los diversos colores básicos en el Spectrum, corresponden a la suma de los fundamentales mediante los cuales se obtienen.

*

La instrucción INVERSE 1 permite intercambiar los colores de fondo y primer término de cualquier carácter.

CARACTERES DE CONTROL

FUNCIÓN	CARACTER	FUNCIÓN	CARACTER
INK	CHR\$ 18	PAPER	CHR\$ 17
FLASH	CHR\$ 19	BRIGHT	CHR\$ 19
INVERSE	CHR\$ 20	OVER	CHR\$ 21

tenza sucesivos serán esentos en el color inicial (el negro, al encender el ordenador), y no en verde.

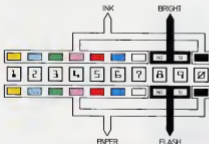
Este mismo resultado puede conseguirse con cualquiera de los otros atributos PAPER, BRIGHT, FLASH, INVERSE y OVER, seguidos de los parámetros habituales, incluyendo los espacios para transparencia y contraste.

Así pues, las palabras BASIC para el control de atributos, se pueden utilizar tanto en forma de palabras clave BASIC como de funciones. En el pri-

mer caso, su efecto será permanente, en el segundo, cesará cuando termine la instrucción PRINT en que se incluyen.

También es posible incluir funciones de color en varios puntos de una instrucción PRINT, con lo cual el efecto de éstas permanecerá durante todas las impresiones realizadas hasta el próximo cambio de atributo temporal. Para comprobarlo, ejecutemos la siguiente instrucción: PRINT INK 4;"A"; TAB 18;"B"; AT 10,0;"C"; INK 2;"D"; PRINT "E".

Mediante la file superior del teclado, es posible observar caracteres de control del color.



LOS CARACTERES DE CONTROL DEL COLOR

Además de esta forma de incluir colores temporales, nuestro ordenador dispone de un par de sistemas más para cumplir el mismo objetivo, ambos basados en los códigos de control del color.

El primero de estos métodos consiste en sustituir la función de asignación de atributos por su carácter correspondiente en el código ASCII, que lógicamente se encuentra en la zona de caracteres de control. Así pues, podemos sustituir INK por CHR\$ 18, PAPER por CHR\$ 17, FLASH por CHR\$ 19, BRIGHT por CHR\$ 19, INVERSE por CHR\$ 20 y OVER por CHR\$ 21.

FUNCIONES DE IMPRESION

FUNCIÓN	CARACTERES CONTROL	FUNCIÓN	CARACTERES CONTROL
BRIGHT 0	CHR\$ 19-CHR\$ 0	BRIGHT 1	CHR\$ 19-CHR\$ 1
FLASH 0	CHR\$ 18-CHR\$ 0	FLASH 1	CHR\$ 18-CHR\$ 1
INVERSE 0	CHR\$ 20-CHR\$ 0	INVERSE 1	CHR\$ 20-CHR\$ 1

El código habrá de ir seguido por un carácter indicativo del parámetro a utilizar, desde **CHR 0** hasta **CHR 9**, que puede ir precedido con el atributo correspondiente, ya sea mediante un punto y coma (;) o mediante un operador suma (+). De esta forma, **PRINT INK 4...**, es lo mismo que escribir **PRINT CHR 16,CHR 4...**, o bien **PRINT CHR 16+CHR 4...**

Existe aún una última posibilidad para incluir caracteres de control temporales: esta vez directamente desde el teclado. Ello nos va a permitir desarrollar el color, además de en las salidas impresas del programa, cosa que ya hemos hecho hasta ahora por el sistema convencional, dentro en las propias líneas de programa, para resaltarlas de un modo especial al efectuar un listado, realizar comentarios etc...

Para tener acceso a estos códigos de control es necesario situarse en el modo «atendido» (cursor E que se obtiene presionando simultáneamente las teclas **CAPS SHIFT** y **SYMBOL SHIFT**), y emplear los caracteres de la fila superior del teclado (los numéricos). Hecho esto, los dígitos del 0 al 7 establecen el color de papel correspondiente y, si se pulsán en combinación con **CAPS SHIFT**, el color de la tinta.

Continuando en el modo «atendido», podemos tener también acceso a las funciones de brillo e impresión intermitente. Pulsando **B** obtenemos el



Cualquier carácter de control introducido por el teclado es doble, y aunque no se encuentre representado, previa de dos pulsaciones de **DELETE (CAPS SHIFT + 0)** para su eliminación.

brillo normal, y con **B** el brillo especial. De forma similar, con **CAPS SHIFT 8** activamos la intermitencia y con **CAPS SHIFT 9** la desconectamos. Como podemos observar, los códigos a los cuales no se tiene acceso por este sistema, son los especiales: el de transparencia y el de contraste. Además de todo esto, puede también accederse al vídeo inverso, esta vez en el modo usual (cursores K, C e LL) con **CAPS SHIFT + 4**, y desconectarlo a través de **CAPS SHIFT + 3**.

Todos estos caracteres de control son dobles, debido a lo cual, cuando es preciso eliminarlos con **DELETE (CAPS SHIFT+0)** hemos de pulsar la tecla dos veces. Por este mismo motivo, y dado que estos controles no son imprimibles, podemos notar ciertas anomalías en el desplazamiento del cursor sobre una línea con caracteres de control; concretamente, escuchar el «click» de pulsación de la tecla, pero no apreciar el avance del cursor. Como resumen final de todo lo dicho, será de utilidad un cuadro donde se describen las funciones de la fila superior del teclado, en los diferentes modos del cursor.



(Cuadro 4)

Los caracteres de control de color obtenibles mediante el teclado, también se reflejan en los listados de los programas.

●	10 PRINT	●
●	20 REM	●
●	30 INPUT A	●
●	40 PRINT A	●
●	50 REM	●
●	60 REM	●
●	70 PRINT	●
●	80 PRINT "FIN"	●

CONTROL POR EL TECLADO

MODO EXTENDIDO

SIN CAPS SHIFT

0 PAPER 0
1 PAPER 1
2 PAPER 2
3 PAPER 3
4 PAPER 4
5 PAPER 5
6 PAPER 6
7 PAPER 7
9 BRIGHT 1

CON CAPS SHIFT

0 INK 0
1 INK 1
2 INK 2
3 INK 3
4 INK 4
5 INK 5
6 INK 6
8 FLASH 0
9 FLASH 1

MODOS K, L Y C

CAPS SHIFT + 4 = INVERSE 1
CAPS SHIFT + 3 = INVERSE 0

LOGICA INFORMATICA



000 Sistema Ordenador está compuesto fundamentalmente por dos elementos: el **hardware** y el **software**.

El primero de ellos está conformado por el conjunto de elementos físicos que integran la máquina, tales como resistencias, transistores, memorias, microprocesador, etc. De hecho, el término inglés **hardware**, cuya traducción aproximada es «artículo de ferretería», designa de una manera un tanto despectiva a este potente conjunto de instrumentos electrónicos. Así pues, podemos decir que este elemento del Sistema constituye el **SOPORTE FISICO** del mismo.

No obstante, pese a la complejidad técnica de los integrantes del **hardware**, éste carece de utilidad alguna, y se convierte en una mera masa de hi-

los conductores, si no se le brinda un sistema de trabajo que le enseñe a realizar las tareas que deseamos, para las cuales está capacitado. Esto es algo que podríamos llamar **SOPORTE LOGICO**, y constituye el denominado **software**. Término que carece de un significado inglés concreto, y que en realidad nace de una forma un tanto chistosa puesto que el **hardware** es el soporte «duro» (*hard* en inglés significa duro), su complemento ideal será el soporte «blando» o **software** (el significado literal de *soft* es blando).

A continuación analizaremos el funcionamiento interno, del **hardware**, pero para ello necesitamos unas nociones previas de la lógica empleada en la informática.

LOGICA MATEMATICA



hardware



software

El soporte físico del ordenador se denomina hardware (artículo de ferretería); por oposición, el soporte lógico pasó a denominarse software.

Aunque los dos posibles valores lógicos se representan mediante los números 1 y 0, no es conveniente su valor numérico, sino el significado lógico que conllevan.



Cuando construimos un sencillo sistema eléctrico, por ejemplo para encender una bombilla, nuestro problema se reduce a localizar unos elementos bastante comunes, que carecen de complejidad alguna: un trazo de cable, una bombilla, una pila y, sobre todo, un interruptor, que será el que nos permita encender o apagar la bombilla a nuestro gusto.

No obstante, la finalidad de un ordenador es extraordinariamente más compleja, y precisa de unos elementos que le permitan, nada más y nada menos, tratar de una forma **LOGICA** los datos, esto es una incipiente forma de inteligencia, y por tanto, nada sencillo de conseguir.

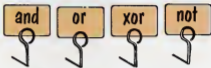
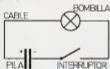
En el año 1847 el matemático inglés George Boole creó una forma particular de las matemáticas, la cual permite el tratamiento de la lógica, denominada **ALGEBRA DE BOOLE**. Aunque en un principio no tuvo una gran difusión, a partir de 1938 comenzó su utilización a gran escala, a raíz de su aplicación por parte de Claude Shannon, en el análisis de redes multiconductos para sistemas telefónicos.

El **ALGEBRA DE BOOLE** es de vital importancia para la informática, puesto que permite el trata-

mento matemático, y por tanto mediante circuitos electrónicos, de la lógica.

La lógica matemática se basa en el sistema binario, en el cual existen dos únicos dígitos: el cero y el uno. Para nosotros, cada uno de estos dígitos

En un esquema eléctrico simple como el del dibujo, el interruptor es el que nos permite determinar el estado de la bombilla.



Las operaciones lógicas que estudiaremos son AND, OR, XOR y NOT.

los representa un ESTADO LÓGICO diferente y opuesto.

El «impulso vital» lo recibe el ordenador a través de la corriente eléctrica, y ésta se puede considerar también bajo un prisma binario: en determinado punto de un circuito hay corriente o no hay corriente, una bombilla está encendida o está apagada, etc.

Así pues, nada más fácil que estudiar los dos estados de la corriente eléctrica (hay o no hay) mediante el álgebra de Boole, destinada precisamente al tratamiento de dispositivos de este tipo binarios o basados.

INFORMACION LÓGICA

Todos los dispositivos básicos del ordenador están preparados para tratar la información binaria: una información que únicamente puede ser un uno o un cero.

Ya hemos visto anteriormente, que si comenzamos a pensar en problemas concretos que poder tratar mediante el sistema binario, llegaremos a la conclusión de que algunos de ellos sí pueden ser representados por este método. Así por ejemplo, dos informaciones opuestas (uno y cero) son suficientes para contener una respuesta afirmativa

BITS

Todo sistema ordenador está compuesto por dos elementos fundamentales: el hardware y el software.

*

En el álgebra de Boole no operamos con valores numéricos, sino con valores lógicos. Los valores pueden adoptar dos únicos estados.

*

Los dos estados lógicos se representan mediante los dígitos 1 y 0.

BITS

En una operación lógica intervienen las variables lógicas (que pueden) y la operación lógica (AND, OR, XOR o NOT), de forma que proporcionan un resultado lógico.



Al valor lógico uno (1) se le suele atribuir un significado positivo: el encendido de una bombilla, etc. El cero recibe siempre el valor opuesto.

$2 \times 1 = 2$	
$2 \times 2 = 4$	
$2 \times 3 = 6$	
$2 \times 4 = 8$	$0 \cdot 0 = 0$
$2 \times 5 = 10$	$1 \cdot 0 = 0$
$2 \times 6 = 12$	$0 \cdot 1 = 0$
$2 \times 7 = 14$	$1 \cdot 1 = 1$
$2 \times 8 = 16$	
$2 \times 9 = 18$	
$2 \times 10 = 20$	

Dentro de la matemática lógica, los tablos de verdad son comparables a nuestras tablas para operaciones aritméticas.

tivó o negativa, si una bombilla está encendida o apagada, etc.

Ahora bien, la información que nosotros deseamos tratar es considerablemente más compleja, y no se reduce a «blanco o negro». Por tanto, agre-



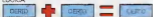
Debido a la similitud entre los resultados lógicos obtenidos por AND, y los que podrían obtenerse considerando sus operaciones como valores numéricos, este operación se conoce como PRODUCTO LÓGICO.

Mientras que las operaciones aritméticas nos proporcionan resultados cuantificados, las operaciones lógicas nos reportan resultados lógicos.

CLÁSICA



LOGICA



mos las unidades simples de información (bits), en conjuntos con más capacidad denominados bytes, y palabras (dos bytes). Tengamos mucho cuidado en no confundir el término informático palabra, con lo que los humanos entendemos como tal en nuestro lenguaje.

Bien, puesto que conocemos el sistema de estudiar informaciones de relativa complejidad, en base al tratamiento de simples unos y ceros, si conseguimos este último, tendremos a nuestro alcance la posibilidad del tratamiento lógico de datos. A partir de este punto es donde entra el álgebra de Boole.

ALGEBRA DE BOOLE

Este instrumento matemático nos permite realizar operaciones entre valores para obtener un resultado, de forma similar a como con nuestras matemáticas de andar por casa hacemos una suma para hallar el total de nuestros ahorros. Ahora bien, entre este tipo de operaciones a las cuales estamos acostumbrados y las que se contemplan en el álgebra de Boole, existen ciertas diferencias.

En primer lugar, tanto los datos a operar como los resultados tienen dos únicos valores posibles: cero o uno. En segundo lugar, las operaciones no están destinadas a obtener resultados de cantidad (total ahorrado), sino lógicos.

Asociada a esta segunda diferencia está la tercera: dado que los resultados a obtener son resultados lógicos, los valores que se operan (cero o uno) no tienen el sentido de cantidad numérica, sino de estado lógico.

Usualmente, para el estado lógico uno (1) adoptamos el significado del cumplimiento de una condición, la existencia de corriente, la bombilla encendida, la respuesta sí, etc. y el cero (0), al no cumplimiento de la condición, la no existencia de corriente, la bombilla apagada, la respuesta no, etc.

Las operaciones del álgebra de Boole que estudiamos se denominan AND (en inglés) y, OR (o), XOR (o exclusivo) y NOT (no). En cuanto a los valores que se operan para obtener el resultado, tomen el nombre de VARIABLES LÓGICAS, y suelen estar representadas por las letras del abecedario (variable lógica A, variable lógica B, etc.), y por último, el resultado de la operación lógica recibe el nombre de RESULTADO LÓGICO.

Del mismo modo que existen unas tablas de sumar, multiplicar, etc., el álgebra de Boole ten-

bién dispone de unas tablas de resultados de sus operaciones, denominadas TABLAS DE VERDAD. Así por ejemplo, si deseamos saber el resultado de la operación 3 por 2, consultamos la tabla de multiplicar y constatamos que es seis. Análogamente, si deseamos conocer el resultado de 1 AND 0, consultamos la tabla de verdad de la operación AND, y averiguamos que su resultado es cero.

Afortunadamente, las tablas de verdad son mucho más simples de aprender que las de operaciones convencionales, puesto que sólo utilizan ceros y unos, y no basta más que cuatro combinaciones.

A continuación estudiaremos cada una de las operaciones separadamente.

AND

Se representa mediante un círculo relleno (●). En la siguiente tabla de verdad hemos representado con A y B dos variables lógicas cualesquiera y mediante Q el resultado de la operación en cada caso.

A ● B	Q
0 ● 0	0
1 ● 0	0
0 ● 1	0
1 ● 1	1

La operación AND también es conocida bajo el nombre de PRODUCTO LÓGICO, debido a que el resultado de la operación coincide enteramente con los valores numéricos de una multiplicación clásica.

El hecho de que recibe el nombre de AND (en inglés Y) no es una mera coincidencia. Realmente, AND es el operador lógico que resuelve problemas del tipo Si <condición> Y <condición> ENTONCES ... <resultado>.

Veamos un ejemplo. Supongamos que deseamos cruzar un paso a nivel con dos barreras (una a cada lado de la vía), el que podamos pasar o no (resultado lógico) depende directamente de dos circunstancias (variables lógicas) que se encuentren levantada la barrera uno Y (AND) que se encuentre levantado la barrera dos.

Llamaremos A a la variable lógica "la barrera uno está levantada", que naturalmente puede adoptar dos valores: 1 (si es cierto que la barrera uno está levantada) ó 0 (no es verdad que la barrera



Para que podamos cruzar un paso a nivel con dos barreras, se tiene que cumplir una estructura lógica del tipo AND.

uno está levantada) ó 0 será la otra variable lógica, correspondiente a "la barrera dos está levantada", que como en el caso anterior adoptará los valores 1 ó 0, según la variable sea cierta o no. El resultado de nuestro problema, denominado Q, será "podemos cruzar la vía", y como es natural, adoptará también dos valores: 1 si podemos cruzar, y 0 si no podemos. Ahora debemos encontrar la operación a utilizar. Puesto que es necesario que las variables A y B se cumplan para que se cumpla el resultado, nos encontramos ante una operación AND. Ya sólo nos queda dar valores a las variables y averiguar el resultado en cada caso.

Si la barrera uno no está levantada Y la barrera dos no está levantada, o lo que es lo mismo: A vale cero y B vale cero. Consultando en la tabla de verdad observaremos que 0 ● 0 = 0. Por lo tanto, no podemos pasar.

Si la barrera uno está levantada Y la barrera dos no está levantada, es decir, A vale uno Y B vale cero. Según la tabla de verdad 1 ● 0 = 0. Así pues, no podemos pasar. Sería también quedarnos entre las dos barreras y que entonces viniera el tren.

Si la barrera uno no está levantada Y la barrera dos está levantada, lo cual se puede expresar también de la siguiente forma: A vale cero Y B vale uno. Consultando en la tabla de verdad observaremos que 0 ● 1 = 0. Por lo tanto, no podemos pasar.

Si la barrera uno está levantada Y la barrera dos está levantada, lo cual significa que A vale uno Y B vale uno. Siguiendo la tabla de verdad diremos que 1 ● 1 = 1. Es decir, podemos pasar puesto que ninguna de las barreras nos lo impide.



!

El **hardware**, término inglés cuyo significado es variado de lo técnico, constituye el soporte físico del ordenador: su carcasa, sus memorias, sus tensiones y todos sus demás componentes.

*

El término inglés **software**, derivado de un significado literal, y que se usó por oposición a **hardware** (hard significa duro y soft, blando), designa el soporte lógico del ordenador: el conjunto de los programas. Quien así permite convertir el hardware en un sistema ordenador.

*

La lógica empleada en la informática se rige por las reglas matemáticas enunciadas por George Boole, conocidas como **ALGEBRA DE BOOLE**.

BIORRITMOS



OY es uno de esos días en que más nos hubiera valido no levantarnos de la cama. Nos duele la cabeza, los digestos se suceden, y por si fuera poco, la moral no nos acompaña. Este es el momento de consultar nuestros biorritmos para comprobar si la situación es pasajera, o realmente la mala fortuna se ceba en nosotros.

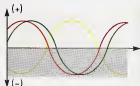
Desde el mismo momento de nuestro nacimiento, cuando el cuerpo humano comienza a vivir con independencia, se inician una serie de ciclos vitales que nos acompañarán durante toda nuestra existencia. Todos estos ciclos se repiten en determinados periodos de tiempo fijos, siguiendo siempre unos mismos índices de ascenso y caída, de forma que una vez conocido su punto inicial (el día de nacimiento), y el tiempo transcurrido desde ese momento, podemos calcular el punto exacto en que se encuentra cualquiera de los ciclos.

Los biorritmos se basan fundamentalmente en tres ciclos vitales: ciclo físico, ciclo emocional y ciclo intelectual. Todos ellos, como hemos dicho anteriormente, tienen su origen común en el día de nuestro nacimiento, y su duración exacta es la siguiente:

- Ciclo físico - 23 días
- Ciclo emocional - 28 días
- Ciclo intelectual - 33 días

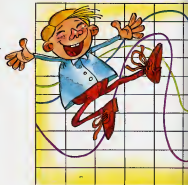
Los puntos críticos son los más importantes dentro de la desorganización de los biorritmos, y corresponden a la intersección de las curvas con el eje central.

PUNTOS CRÍTICOS



En la zona negativa del biorritmo, nos encontramos en un mal momento para acometer tareas en el área representada.

Esto quiere decir, que tomando como punto de partida (nacimiento) el cero, y estudiando, por ejemplo, el ciclo físico, observaremos que se produce una pendiente de ascenso cuya duración es siete días, la cual va seguida de un descenso de catorce días, a cuyo mitad volveremos a atravesar el punto cero, pero en esta ocasión descendiendo, por último, el ciclo se cerrará volviendo



BITS

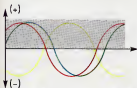
La introducción de fechas ha de ser como cada uno de los ejemplares que aparecen en la pantalla.



El año mismo que editó el programa es el 6666.

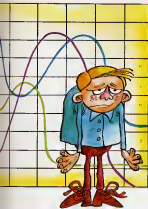


A los puntos de intersección de las curvas con el eje central se les denominan PUNTOS CRÍTICOS.



En la zona positiva de las curvas, nuestra productividad hacia los fines que representen es buena

a un ascenso, que en los siete días restantes nos quedarán de nuevo en la posición de inicio. Técnicamente, podemos decir que la representación del ritmo biológico está determinada por una sinusoidal, de amplitud comprendida entre los valores +1 y -1, correspondientes a la fórmula: $Amplitud = \text{SENO}(360 \times ND / DC)$; donde ND es el número de días después del nacimiento, y DC es la duración de cada ciclo



Cuando la sinusoidal se encuentra en la zona positiva (zona superior de la pantalla), indica una buena predisposición ante el área que señala la curva de un ciclo determinado. Si por el contrario el período de tiempo que nos interesa estuviese encuadrado en la zona inferior que determina el eje horizontal en la pantalla, más nos valdrá andar con cuidado.

Esto no quiere decir que en el caso de una mala racha de nuestro ciclo físico, vayamos a enfermarnos, pero sí es más fácil que nos sintamos cansados o seamos más propensos a alguna infección.

Del mismo modo, atravesar una buena época intelectual, nos afecta en el sentido de reportarnos menor trabajo estudiar un mismo tema, o comprender determinado problema, un estado emocionalmente bajo, nos hace propensos a las depresiones, o convertir un problema que podía ser un gránulo de arena en una auténtica montaña. Las intenciones en el alma, se decantan fundamentalmente hacia la opción de que más importantes que las zonas positivas o negativas de los ciclos, son los denominados puntos críticos aquellos en los cuales la curva atraviesa el límite entre las dos zonas, es decir, se encuentra en el que hemos denominado punto cero.

Durante estos puntos críticos somos especialmente sensibles, dado que nos encontramos en una fase de transición entre unos estados y otros, y por tanto, en una situación de claro desequilibrio. Así pues, será tanto más delicado el momento, cuanto más puntos críticos concurran. Es relativamente frecuente que dos ciclos atraviesen un punto crítico el mismo día, en cuyo caso nuestro estado de desequilibrio puede jugarlos una mala pasada.

No obstante, la coincidencia de tres puntos críticos, cosa por otra parte bastante clara, puesto que en el preciso momento de nacer atravesamos el punto más crítico de nuestro vida, no se repite hasta cincuenta y siete años más tarde. La teoría de los berberinos es seguida desde hace varios años por distintas entidades de países tan desarrollados como Japón, Estados Unidos, Rusia e incluso España, para mejorar la eficacia en el trabajo.

En este sentido, es ya conocido el caso de una compañía de transportes japonesa, la cual averiguó mediante un estudio estadístico, que los accidentes sufridos por sus conductores solían

BITS

Los caracteres que aparecen en el listado subrayados doblemente, deben introducirse como los gráficos cambiados (con CAPS SHIFT) de los teclados correspondientes.



Los caracteres subrayados INV y TRUE, pertenecen entre comillas, al código de control INVERSE VIDEO y TRUE VIDEO, obtenidos por el teclado mediante CAPS SHIFT + 4 y CAPS SHIFT + 3, respectivamente.



Para la grabación del programa tendremos el siguiente comando: **SAVE "BIO"**. En el caso de que lo deseáramos almacenar con subsecuencia, utilizaremos la siguiente instrucción: **SAVE "BIO" LINE 10**.

